Universidad Rey Juan Carlos

E.T.S. Ingeniería de Telecomunicación

Redes de Ordenadores para Robots y Máquinas Inteligentes

Práctica 3

Autor:
Javier Izquierdo Hernández

February 24, 2024

Contenidos

Antes de comenzar, descarga tu escenario del siguiente enlace donde deberás introducir tu número de DNI (8 dígitos) con la letra correspondiente:

https://mobiquo.gsyc.urjc.es/practicas/ror/p3.html

En la figura 1 se representa un conjunto de subredes y máquinas (pc1, pc2, pc4, pc5, r1, r2 y firewall) que pertenecen a una determinada empresa y su conexión a Internet a través de la máquina firewall. La empresa tiene definidas un conjunto de subredes de ámbito privado:

- 10.X.0.0/24: r1(eth1), pc1, pc2
- 10.X.1.0/24: firewall(eth0), r1(eth0), r2(eth0)
- 10.X.2.0/24: r2(eth1), pc3

Adicionalmente, la empresa tiene las máquinas pc4 y pc5 que se encuentran en una subred pública: 100.X.0.0/24. Estas máquinas proporcionan servicios básicos de la empresa: servidor de HTTP y servidor de fecha y hora. A esta zona de la red interna donde la empresa tiene una o varias subredes públicas para ofrecer servicios a Internet se le denomina zona desmilitarizada o DMZ (DeMilitarized Zone).

Todas las máquinas de la empresa se conectan a Internet a través de la máquina firewall y la subred 100.X.1.0/24.

En este escenario, se considera que Internet está formado por las siguientes máquinas: r3, r4, r5, pc6 y pc7 que se encuentran conectadas a las siguientes subredes públicas:

- 100.X.1.0/24: r3(eth0)
- 100.X.2.0/24: r3(eth1), r5(eth2)
- 100.X.3.0/24: r3(eth2), r4(eth2)
- 100.X.4.0/24: r4(eth1), r5(eth0)
- 100.X.5.0/24: r4(eth0), pc6
- 100.X.6.0/24: r5(eth1), pc7

Arranca de una en una todas las máquinas de la figura.

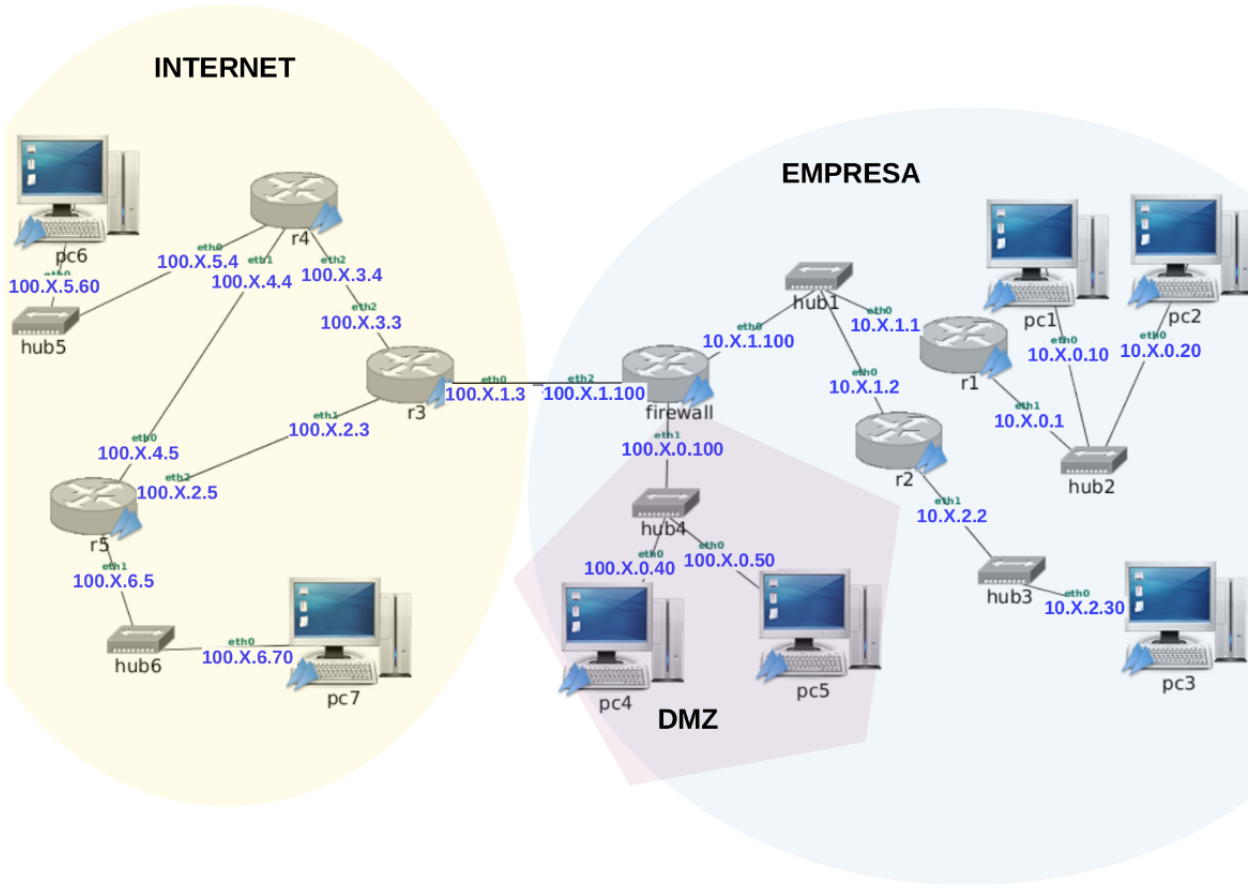


Figure 1: Escenario de red para la configuración de un firewall

1 Introducción

A continuación se proporcionan algunos consejos para facilitar la realización de la práctica.

1.1 Edición y ejecución de scripts

En esta práctica se configurará la máquina firewall para que actúe como traductor de direcciones y como cortafuegos. Habrá que definir varias reglas utilizando iptables. Por este motivo, es recomendable guardar dichas reglas en un fichero script de shell.

Considera la posibilidad de editar y guardar el script en el sistema de ficheros de la máquina real, ejecutándolo desde dentro de la máquina virtual. Así, si tu script fw.sh está almacenado directamente en tu HOME de la máquina real, podrías editarlo en ella con un editor gráfico (por ejemplo, gedit) y luego ejecutarlo en la máquina firewall escribiendo dentro de esa máquina virtual:

/hosthome/fw.sh

1.2 Comprobación de la configuración del *firewall*

Durante la práctica frecuentemente tendrás que ir comprobando que el firewall está correctamente configurado, es decir:

- deja pasar el tráfico que está permitido.
- impide el paso del tráfico que debe ser bloqueado.
- realiza la traducción de direcciones IP necesaria para que no aparezcan en Internet paquetes con direcciones privadas

Para ello deberás emplear la herramienta netcat (nc) (ya utilizada en prácticas del curso pasado) que permite arrancar aplicaciones TCP y UDP en modo cliente o servidor.

El enunciado de la práctica te irá indicando cuándo y en qué máquinas debes lanzar un cliente o un servidor TCP o UDP para ir probando la configuración del firewall. Consulta la documentación adjunta para recordar la sintaxis de netcat.

2 Traducción de direcciones y puertos en el firewall: tabla nat

2.1 Clientes en la red privada, servidores externos

Configura un script **fw1.sh** en el *firewall* para que:

- se borren las reglas que hubiera configuradas previamente en la tabla **nat**
- se reinicien los contadores de la tabla **nat**
- se realice la traducción de direcciones para el tráfico saliente de las redes privadas (SNAT) y su correspondiente tráfico de respuesta.

Incluye el script en la memoria. Ver el script en 4.1.

2.1.1 Pruebas con TCP

Ejecuta el script **fw1.sh** de 2.1.

1. Captura el tráfico en **r3-eth0** (iptables-01.cap) y en **firewall-eth0** (iptables-02.cap) para ver los paquetes dentro de la red de la Empresa y por Internet. Arranca las siguientes aplicaciones:

   - nc como servidor TCP en pc6, puerto 7777
   - nc como cliente TCP en pc1

   Sin escribir nada ni en el cliente ni en el servidor, consulta la información de **ip_conntrack** del **firewall** cada medio segundo. Para hacerlo automáticamente, en vez de repetir el comando utiliza watch de la siguiente forma:

   firewall:~# watch -n 0.5 cat /proc/net/ip_conntrack

Explica el número de paquetes que se han observado en cada sentido, razonando la respuesta, indicando de qué paquetes se trata (recuerda que estamos ante una conexión TCP).

Se han mandado dos mensajes desde pc1 a pc6, el mensaje de Syn para abrir la conexión y el mensaje de Ack a la respuesta al primer mensaje de pc6.
Y en la dirección contraria se habrá mandado 1 solo mensaje que será la respuesta a la apertura de la conexión.

```
tcp      6 431988 ESTABLISHED src=10.209.0.10 dst=100.209.5.60 sport=6666 dport=7777 packets=2 bytes=112
 src=100.209.5.60 dst=100.209.1.100 sport=7777 dport=6666 packets=1 bytes=60 [ASSURED] mark=0 use=1
```

2. Introduce una palabra en la entrada estándar de pc1, pulsa <Enter> y observa los cambios en **ip_conntrack**. Explica a qué se deben.

   El tiempo que le queda a la conexión se resetea a 120 horas o 432000 segundos porque se recibe un paquete.

```
tcp      6 431999 ESTABLISHED src=10.209.0.10 dst=100.209.5.60 sport=6666 dport=7777 packets=4 bytes=222 src=10
0.209.5.60 dst=100.209.1.100 sport=7777 dport=6666 packets=3 bytes=164 [ASSURED] mark=0 use=1
```

3. Realiza un Ctrl+C en el terminal de **pc1** para interrumpir la ejecución de **nc**. Observa los cambios en **ip_conntrack** y explica a qué se deben.

   El tiempo que le queda a la conexión desciende a 120 segundos y la conexión pasa a tener el valor de TIME_WAIT ya que esa conexión ha sido cerrada tanto por el cliente como por el servidor, entonces no tiene sentido recordarla.

```
tcp      6 119 TIME_WAIT src=10.209.0.10 dst=100.209.5.60 sport=6666 dport=7777 packets=4 bytes=216 src=100.209
.5.60 dst=100.209.1.100 sport=7777 dport=6666 packets=2 bytes=112 [ASSURED] mark=0 use=1
```

4. Interrumpe las capturas, y estúdialas. En particular, identifica los mismos paquetes en las 2 capturas, y observa cómo cambian las direcciones IP de los mismos paquetes según viajen dentro de la EMPRESA o por INTERNET. Explica el resultado.

   En la captura dentro de la empresa en la interfaz eth0 del firewall se puede ver que los mensajes van de la dirección Ip de pc1 a la de pc6 y viceversa, sin ninguna traducción.
   Mientras que en la captura en Internet los mensajes se intercambian entre pc6

y la dirección Ip del firewall que da con Internet, es decir, que la dirección Ip de pc2 no aparece en ningún momento.

5. Consulta la lista de reglas en el firewall con:

   firewall:~# iptables -t nat -L -v -n

   Obseva qué regla(s) están cumpliendo los paquetes y cuántas veces se cumple(n).

   Se está cumpliendo la única regla que hemos añadido

   - iptables -t nat -A POSTROUTING -s 10.209.0.0/16 -o eth2 -j SNAT –to-source 100.209.1.100

   1 vez.

```
firewall:~# iptables -t nat -L -v -n
Chain PREROUTING (policy ACCEPT 1 packets, 60 bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain POSTROUTING (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
    1    60 SNAT       all  --  *      eth2    10.209.0.0/16        0.0.0.0/0           to:100.209.1.100

Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
```

6. Vuelve a repetir la misma prueba anterior (sin necesidad de realizar las capturas de tráfico): lanza servidor y cliente, intercambia tráfico, y termina la conexión. Vuelve a mirar qué regla(s) se están cumpliendo y **cuántas veces** se cumple(n).

   Se cumple 2 veces la misma regla que en el apartado anterior, el de la primera captura y el del inicio de la segunda conexión.

```
firewall:~# iptables -t nat -L -v -n
Chain PREROUTING (policy ACCEPT 2 packets, 120 bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain POSTROUTING (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
    2   120 SNAT       all  --  *      eth2    10.209.0.0/16        0.0.0.0/0           to:100.209.1.100

Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
```

2.1.2 Pruebas con UDP

Ejecuta el script **fw1.sh** de 2.1 para que se reinicien los contadores de paquetes de iptables, compruébalo consultando la lista de reglas del firewall.

1. Captura el tráfico en **r3-eth0** (iptables-03.cap) y en **firewall-eth0** (iptables-04.cap) para ver los paquetes dentro de la red de la Empresa y por Internet. Arranca las siguientes aplicaciones:

   - **nc** como servidor UDP en **pc6**, puerto 7777
   - **nc** como cliente UDP en **pc2**

   Realiza las siguientes pruebas:

   a) Sin escribir nada ni en el cliente ni en el servidor, consulta la información de **ip_conntrack** del **firewall** cada medio segundo. Recuerda que el tráfico es ahora UDP y no hay conexiones propiamente dichas. Explica el resultado.
   No hay nada, porque al no escribir no se inicia una conexión.

   b) Escribe 5 líneas en el terminal de **pc2** para que se las envíe a **pc6**. Explica el número de paquetes enviados en la información que muestra **ip_conntrack**.
   Ip_conntrack muestra 5 paquetes.

```
udp      17 26 src=10.209.0.20 dst=100.209.5.60 sport=2222 dport=7777 packets=5 bytes=150 [UNREPLIED] src=100.2
09.5.60 dst=100.209.1.100 sport=7777 dport=2222 packets=0 bytes=0 mark=0 use=1
```

   c) Escribe una línea en **pc6** para que se la envíe a **pc2**. Explica nuevamente el número de paquetes en **ip_conntrack**.

   Si pc6 envía el mensaje antes de que pasen los 30 segundos necesarios para que el firewall olvide la asociación aparecerán 5 paquetes como anteriormente en la "primera" parte, mientras que aparecerá un paquete en la otra.
   Si pasan los 30 segundos el servidor se cerrará al dar error en la conexión.

```
udp      17 11 src=10.209.0.20 dst=100.209.5.60 sport=6666 dport=7777 packets=5 bytes=150 src=100.209.5.60 dst=
100.209.1.100 sport=7777 dport=6666 packets=1 bytes=30 mark=0 use=1
```

   d) Observa el poco tiempo que se mantiene la "asociación" entre cliente y servidor en **ip_contrack**. Indica cuánto ha sido.
   Han sido 30 segundos.

e) Interrumpe la captura y las ejecuciones de **nc**, explica la captura y cómo ésta se relaciona con la información que has visto en **ip_conntrack**.

En r3 se puede ver que los datagramas Udp tienen la Ip de origen como 100.209.1.100 que es la del firewall al igual que en la segunda línea que se ve en la imagen superior en ip_conntrack. Mientras que en eth0 del firewall se pueden ver que los mensajes serán entre las direcciones Ip mostradaas en la primera línea.
También se aprecia que han habido 5 mensajes desde pc2 a pc6 y que pc6 ha contestado a pc2 1 vez, esto se ve en ip_conntrack en el número de paquetes en cada "sentido".

2. Consulta la lista de reglas en el **firewall**, e indica cuáles se están cumpliendo y cuántas veces se cumplen.

Se está cumpliendo la única regla que hemos añadido

- iptables -t nat -A POSTROUTING -s 10.209.0.0/16 -o eth2 -j SNAT –to-source 100.209.1.100

1 vez.

```
firewall:~# iptables -t nat -L -v -n
Chain PREROUTING (policy ACCEPT 1 packets, 30 bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain POSTROUTING (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
    1    30 SNAT       all  --  *      eth2    10.209.0.0/16        0.0.0.0/0           to:100.209.1.100

Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
```

3. Interrumpe la ejecución de cliente y servidor e inicia una nueva comunicación entre un nuevo cliente y un servidor UDP e intercambia tráfico entre ellos para ver cómo evolucionan las cuentas en la lista de reglas. Explica qué reglas se están cumpliendo ahora y **cuántas veces** se cumplen.

Se cumple la misma regla que en el anterior y el número de veces dependerá de cada cuanto se mandes los mensajes desde pc2, ya que si se mandan antes de los 30 segundos no contarán, pero en caso contrario incrementará en 1 el contador.

4. Captura de nuevo el tráfico en **r3-eth0** (iptables-05.cap) y en **firewall-eth0** (iptables-06.cap) para ver los paquetes dentro de la red de la Empresa y

```
firewall:~# iptables -t nat -L -v -n
Chain PREROUTING (policy ACCEPT 3 packets, 94 bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain POSTROUTING (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
    2    64 SNAT       all  --  *      eth2    10.209.0.0/16        0.0.0.0/0           to:100.209.1.100

Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
```

por Internet cuando tienes varios clientes desde un mismo puerto origen conectándose a un mismo servidor, para ello inicia:

- **nc** como servidor UDP en **pc7**, puerto 7777
- **nc** como cliente UDP en **pc1**, puerto 6666
- **nc** como cliente UDP en **pc2**, puerto 6666

Ahora, envía una línea desde **pc1** y después una línea desde **pc2**. Ten en cuenta que **nc** no funciona como las aplicaciones servidoras que pueden atender a varios clientes a la vez. La aplicación **nc** no está preparada para que un servidor se pueda comunicar a la vez con dos clientes, por ello el envío desde **pc2** provocará que **pc7** envíe un ICMP de error a **pc2**. Pero para lo que queremos comprobar este error no es importante, sólo queremos analizar lo que ocurre en el **firewall** con la traducción de direcciones IP y puertos.

Interrumpe las capturas y analízalas fijándote en las direcciones IP y **puertos** que se utilizan en la red de la EMPRESA y en INTERNET.

En la interfaz eth0 del firewall se reciben los 2 mensajes udp de pc1 y pc2 con sus Ip originales como origen y la Ip de pc7 como destino, y también con sus puertos originales. También esta presente el datagrama Icmp que envía pc7 al dar error, y este mantiene los mismo puertos que el mensaje original de pc2.
Mientras tanto en r3 se puede ver que las Ip son reemplazadas por la del router que da a Internet y que el puerto del mensaje de pc2 ha cambiado a 1024 en vez de 6666, ya que este coincidía con el que tenía asignado anteriormente pc1. Lo mismo pasa para el mensaje de error, que va dirigido a la Ip del router y el puerto 1024.

2.1.3 Pruebas con ICMP

Ejecuta el script **fw1.sh** de 2.1 para que se reinicien los contadores de paquetes de iptables. Vamos a generar tráfico ICMP.

1. Realiza una captura en **pc6** (iptables-07.cap) y otra en **r1(eth1)** (iptables-08.cap).

2. Ejecuta el siguiente comando en **pc1** (recuerda sustituir la X por el número que te corresponde):

   pc1:~# ping -c 2 100.X.5.60

3. Interrumpe las capturas anteriores.

4. Consulta la información de **ip_conntrack** del **firewall**. Verás que no aparece nada. Recuerda que esto se debe a que las "conexiones" que se consideran para los paquetes ICMP es una diferente entre cada *echo request* y su correspondiente *echo reply*, asociación que se "olvida" justo después del *echo reply*.

5. Consulta la lista de reglas en el **firewall**, y mira cuáles se están cumpliendo y **cuántas veces**, relaciona esta información con los mensajes capturados.

   Se está cumpliendo la regla

   - iptables -t nat -A POSTROUTING -s 10.209.0.0/16 -o eth2 -j SNAT –to-source 100.209.1.100

   2 veces, ya que al ser una conexión ICMP la asociación se olvida en cuanto llega el *reply*, por lo tanto al mandar 2 *pings* (se ven en ambas capturas) se ha iniciado la asociación 2 veces.

```
firewall:~# iptables -t nat -L -v -n
Chain PREROUTING (policy ACCEPT 2 packets, 168 bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain POSTROUTING (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
    2   168 SNAT       all  --  *      eth2    10.209.0.0/16        0.0.0.0/0           to:100.209.1.100

Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
```

2.2 Servidores en la red privada, clientes externos

Aunque en una red como la que aparece en la figura, lo habitual es colocar los servidores accesibles desde el exterior en la zona DMZ, para ver cómo funciona DNAT, vamos a permitir que haya servidores accesibles desde el exterior en la red privada interna.

2.2.1 Apertura de puertos TCP

Realiza un nuevo script **fw2.sh** en el firewall para que:

- se borren las reglas que hubiera configuradas previamente en la tabla **nat**
- se reinicien los contadores de la tabla **nat**
- el tráfico de entrada al firewall destinado al puerto TCP 80 sea redirigido a **pc3**, puerto 80.

Incluye el script en la memoria. Ver el script en 4.2. Ejecuta dicho script y arranca las siguientes aplicaciones:

- Inicia una captura en **r2(eth1)**(iptables-09.cap) y en **r4(eth0)** (iptables-10.cap).
- **nc** como servidor TCP en **pc3**, puerto 80
- **nc** como cliente TCP en **pc6**, de forma que su tráfico lo reciba el servidor de **pc3** (NOTA: presta especial atención a los parámetros con los que debes lanzar este cliente). Indica en la memoria el comando que has usado para lanzar el cliente y explica por qué lo has hecho así.

  He usado el comando: nc -p 6666 100.209.1.100 80.
  He usado esa Ip porque el firewall redirige automaticamente los mensajes que llegan a esa Ip y puerto a pc3 puerto 80, también porque si se lo envías a la dirección de pc3 los routers de internet no sabrán como redirigirlo.
- Escribe una palabra en el lado cliente y pulsa <Enter>
- Interrumpe la ejecución del cliente y el servidor.
- Interrumpe las capturas.

Explica los siguientes resultados:

1. El resultado observado en **ip_conntrack** y la traducción de direcciones IP y puertos realizada.

   Se puede observar en la imagen inferior que los datagramas enviados a la ip del router (100.209.1.100) con el puerto 80 son redirigidos a pc3 y al puerto 80.

```
tcp      6 431993 ESTABLISHED src=100.209.5.60 dst=100.209.1.100 sport=6666 dport=80 packets=3 bytes=169
 src=10.209.2.30 dst=100.209.5.60 sport=80 dport=6666 packets=2 bytes=112 [ASSURED] mark=0 use=1
```

2. La lista de reglas en el **firewall**, indica cuáles se están cumpliendo y cuántas veces.

   Se está cumpliendo la regla

   - iptables -t nat -A PREROUTING -i eth2 -d 100.209.1.100 -p tcp –dport 80 -j DNAT –to-destination 10.209.2.30:80

   1 vez solo, ya que al ser una conexión Tcp solo se incrementará ese valor si se espera entre mensaje y mensaje 5 días.

```
firewall:~# iptables -t nat -L -v -n
Chain PREROUTING (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
    1    60 DNAT       tcp  --  eth2   *       0.0.0.0/0            100.209.1.100       tcp dpt:80 to:10.209.2.
30:80

Chain POSTROUTING (policy ACCEPT 1 packets, 60 bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
```

3. Relaciona las reglas que se han cumplido con los datos de los mensajes capturados en los ficheros.

   En r2 se puede ver el intercambio de mensajes entre pc6 y pc3 con los datagramas que envía pc6 con la traducción ya hecha.
   Mientras que en r4 parece que el intercambio de mensajes es entre pc6 y el firewall como se puede ver en la siguiente imagen.

```
 1 0.000000    100.209.5.60     100.209.1.100    TCP    74 6666 → 80 [SYN] Seq=0 Win=5840 Len=0 MSS=1460 SACK_PERM TSval=600733 TSecr=0 WS=2
 6 8.992079    100.209.1.100    100.209.5.60     TCP    74 80 → 6666 [SYN, ACK] Seq=0 Ack=1 Win=5792 Len=0 MSS=1460 SACK_PERM TSval=4294942175 TSecr=601633 WS=2
 7 8.992186    100.209.5.60     100.209.1.100    TCP    66 6666 → 80 [ACK] Seq=1 Ack=1 Win=5840 Len=0 TSval=601633 TSecr=4294942175
 8 8.993209    100.209.5.60     100.209.1.100    TCP    71 6666 → 80 [PSH, ACK] Seq=1 Ack=1 Win=5840 Len=5 TSval=601633 TSecr=4294942175
 9 8.994025    100.209.1.100    100.209.5.60     TCP    66 80 → 6666 [ACK] Seq=1 Ack=6 Win=5792 Len=0 TSval=4294942175 TSecr=601633
12 17.123145   100.209.5.60     100.209.1.100    TCP    66 6666 → 80 [FIN, ACK] Seq=6 Ack=1 Win=5840 Len=0 TSval=602446 TSecr=4294942175
13 17.124337   100.209.1.100    100.209.5.60     TCP    66 80 → 6666 [FIN, ACK] Seq=1 Ack=7 Win=5792 Len=0 TSval=4294942988 TSecr=602446
14 17.124442   100.209.5.60     100.209.1.100    TCP    66 6666 → 80 [ACK] Seq=7 Ack=2 Win=5840 Len=0 TSval=602446 TSecr=4294942988
```

2.2.2 Apertura de puertos UDP

Modifica el script **fw2.sh** para que, adicionalmente:

- el tráfico de entrada al firewall destinado al puerto UDP 5001 sea redirigido a **pc1**, puerto 5001
- El tráfico de entrada al firewall destinado al puerto UDP 5002 sea redirigido a **pc2**, puerto 5001

Incluye el script en la memoria. Ejecuta el script que acabas de modificar y arranca las aplicaciones:

- Inicia una captura en **r1(eth1)** (iptables-11.cap), en **r4(eth0)** (iptables-12.cap) y en **r5(eth1)** (iptables-13.cap).
- **nc** como servidor UDP en **pc1**, puerto 5001
- **nc** como servidor UDP en **pc2**, puerto 5001
- **nc** como cliente UDP en **pc6**, de forma que su tráfico lo reciba el servidor de **pc1**. Indica el comando que has utilizado para lanzar el cliente y explica por qué.

  He usado el comando: nc -u -p 6666 100.209.1.100 5001.
  He usado esa Ip porque el firewall redirige automaticamente los mensajes que llegan a esa Ip y puerto a pc1 puerto 5001, también porque si se lo envías a la dirección de pc1 los routers de internet no sabrán como redirigirlo.
- **nc** como cliente UDP en **pc7**, de forma que su tráfico lo reciba el servidor de **pc2**. Indica el comando que has utilizado para lanzar el cliente y explica por qué.

  He usado el comando: nc -u -p 6666 100.209.1.100 5002.
  He usado esa Ip porque el firewall redirige automaticamente los mensajes que llegan a esa Ip y puerto a pc2 puerto 5001, también porque si se lo envías a la dirección de pc2 los routers de internet no sabrán como redirigirlo.
- Escribe una palabra en el lado cliente de **pc6** y **pc7** y pulsa <Enter>
- Interrumpe la ejecución de los clientes y servidores.
- Interrumpe las capturas.

Explica los siguientes resultados:

1. El resultado observado en **ip_conntrack** y la traducción de direcciones IP y puertos realizada.

   Se puede observar en la imagen inferior que los datagramas enviados a la ip del router (100.209.1.100) con el puerto 5001 son redirigidos a pc1 y al puerto 5001, mientras que los que van al puerto 5002 del firewall son redirigidos al puerto 5001 de pc2.

```
udp      17 25 src=100.209.5.60 dst=100.209.1.100 sport=6666 dport=5001 packets=1 bytes=30 [UNREPLIED] s
rc=10.209.0.10 dst=100.209.5.60 sport=5001 dport=6666 packets=0 bytes=0 mark=0 use=1
udp      17 28 src=100.209.6.70 dst=100.209.1.100 sport=6666 dport=5002 packets=1 bytes=30 [UNREPLIED] s
rc=10.209.0.20 dst=100.209.6.70 sport=5001 dport=6666 packets=0 bytes=0 mark=0 use=1
```

2. Consulta la lista de reglas en el **firewall** e indica cuáles se están cumpliendo y cuántas veces.

   Se están cumpliendo las reglas

   - iptables -t nat -A PREROUTING -i eth2 -d 100.209.1.100 -p udp –dport 5001 -j DNAT –to-destination 10.209.0.10:5001
   - iptables -t nat -A PREROUTING -i eth2 -d 100.209.1.100 -p udp –dport 5002 -j DNAT –to-destination 10.209.0.20:5001

   1 vez cada una.

```
firewall:~# iptables -t nat -L -v -n
Chain PREROUTING (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 DNAT       tcp  --  eth2   *       0.0.0.0/0            100.209.1.100       tcp dpt:80 to:10.209.2.30:80
    1    30 DNAT       udp  --  eth2   *       0.0.0.0/0            100.209.1.100       udp dpt:5001 to:10.209.0.10:5001
    1    30 DNAT       udp  --  eth2   *       0.0.0.0/0            100.209.1.100       udp dpt:5002 to:10.209.0.20:5001

Chain POSTROUTING (policy ACCEPT 2 packets, 60 bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
```

3. Relaciona las reglas que se han cumplido con los datos de los mensajes capturados en los ficheros.

   En r1 se pueden ver los 2 mensajes con las direcciones traducidas.

```
3 0.000080      100.209.5.60                    10.209.0.10                     UDP         44 6666 → 5001 Len=2[Malformed Packet]
6 2.067981      100.209.6.70                    10.209.0.20                     UDP         44 6666 → 5001 Len=2[Malformed Packet]
```

   Luego en r4 se ve el datagrama que envía pc6 a la dirección Ip del firewall que da a Internet y con el puerto 5001.
   Y por último en r5 se ve el datagrama que envía pc7 a la dirección Ip del firewall que da a Internet y con el puerto 5002.

3 Filtrado de tráfico en el *firewall*: tabla filter

Crea un *script* **fw3.sh** en el **firewall** partiendo de la configuración de traducción de direcciones realizada en **fw1.sh** (clientes en la red privada, servidores externos) al que se le añada la siguiente configuración (todas en el mismo *script*). Descripción de las **especificaciones**:

1. Reiniciar la tabla **filter**: borrar su contenido y reiniciar sus contadores.

2. Fijar las políticas por defecto de las cadenas de la tabla **filter**, haciendo que por defecto se descarte todo el tráfico en el **firewall** excepto los paquetes que cree el propio **firewall** (configuración habitual en un *firewall*).

3. Permitir el tráfico de entrada dirigido a las aplicaciones que se están ejecutando en el propio **firewall** únicamente si este tráfico tiene su origen en las subredes privadas de la empresa.

4. Permitir todo el tráfico saliente desde las subredes privadas hacia Internet y el tráfico de respuesta al saliente.

   Ten en cuenta que como has partido del *script* **fw1.sh**, en dicho *script* ya tenías las reglas de la tabla **nat** para la traducción de la dirección IP de origen de los paquetes que reenvía el **firewall** y los paquetes del tráfico entrante de respuesta a éste.

5. Permitir desde Internet únicamente el tráfico entrante nuevo hacia la zona DMZ según las siguientes reglas:

   - acceso a un servidor *echo* existente en **pc4** (UDP, puerto 7). El servidor de *echo* es un servidor que al enviarle una cadena de caracteres, devuelve la misma cadena que se le ha enviado. Para comprobar el acceso a este servidor utiliza **nc** como cliente desde otra máquina.

- acceso a un servidor *daytime* existente en **pc5** (UDP, puerto 13). El servidor *daytime* es un servidor que al enviarle algo, devuelve la fecha y hora de la máquina donde está instalado. Para comprobar el acceso a este servidor utiliza **nc** como cliente desde otra máquina.

Para este tipo de tráfico configura además regla/s con **acción LOG** para que cada vez que se permita el tráfico UDP descrito anteriormente, se deje un mensaje en el fichero de LOG del sistema.

6. Permitir únicamente la comunicación entre la red privada y la zona DMZ de la siguiente forma:

   - acceso desde **pc1** a un servidor de *echo* (TCP, puerto 7) existente en **pc4**. En este caso, como todas las máquinas involucradas en la comunicación pertenecen al ámbito privado de la empresa, no es necesario que realices traducción de direcciones. Para este tipo de tráfico configura además regla/s con **acción LOG** para que cada vez que se permita este tráfico TCP, se deje un mensaje en el fichero de LOG del sistema.

7. Desde la zona DMZ **NO** se debe permitir iniciar ninguna comunicación con la red privada ni con el propio **firewall**.

Incluye el script en la memoria. Ver el script en 4.3.

Prueba el resultado de la configuración del script:

1. Sólo aplicaciones de la red privada pueden comunicarse con aplicaciones ejecutándose en el **firewall**:

   - Ejecuta el script de configuración para que se reinicien los contadores.
   - Prueba a lanzar un servidor TCP en el puerto 1111 de la máquina firewall. Lanza un cliente en **pc1** que se comunique con ese servidor. Indica qué reglas del firewall se están cumpliendo.

     Se esta cumpliendo la regla

     iptables -t filter -A INPUT -i eth0 -j ACCEPT

     2 veces, ya que es una comunicación Tcp.
   - Lanza un cliente en **pc6** que trate de comunicarse con ese servidor. Indica qué provoca que ese tráfico sea descartado.

```
firewall:~# iptables -t filter -L -v -n
Chain INPUT (policy DROP 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
    2   112 ACCEPT     all  --  eth0   *       0.0.0.0/0            0.0.0.0/0
    0     0 DROP       all  --  eth1   *       0.0.0.0/0            0.0.0.0/0           state NEW

Chain FORWARD (policy DROP 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 ACCEPT     all  --  eth0   eth2    0.0.0.0/0            0.0.0.0/0
    0     0 ACCEPT     all  --  eth2   eth0    0.0.0.0/0            0.0.0.0/0           state RELATED,ESTABLISHED
    0     0 ACCEPT     udp  --  *      *       0.0.0.0/0            10.209.0.40         udp dpt:7
    0     0 LOG        udp  --  *      *       0.0.0.0/0            10.209.0.40         udp dpt:7 LOG flags 0 level 4 prefix `echo'
    0     0 ACCEPT     udp  --  *      *       0.0.0.0/0            10.209.0.50         udp dpt:13
    0     0 LOG        udp  --  *      *       0.0.0.0/0            10.209.0.50         udp dpt:13 LOG flags 0 level 4 prefix `daytime'
    0     0 ACCEPT     tcp  --  *      *       0.0.0.0/0            10.209.0.40         tcp dpt:7
    0     0 LOG        tcp  --  *      *       0.0.0.0/0            10.209.0.40         tcp dpt:7 LOG flags 0 level 4 prefix `tcp'
    0     0 DROP       all  --  eth1   eth0    0.0.0.0/0            0.0.0.0/0           state NEW

Chain OUTPUT (policy ACCEPT 1 packets, 60 bytes)
 pkts bytes target     prot opt in     out     source               destination
```

Porque pc6 al ser parte de la red no se le permite iniciar la conexión, ya que por defecto todos los mensajes que no cumplen las reglas se descartan.

2. Comunicación desde un cliente de la red privada con un servidor cualquiera en el exterior:

   - Ejecuta el script de configuración para que se reinicien los contadores.
   - Prueba a lanzar un servidor TCP en **pc6** en el puerto 1111. Realiza una captura de tráfico en **r3(eth0)** (iptables-14.cap). Prueba a lanzar un cliente con **nc** desde **pc1** para que se comunique con este servidor.
   - Consulta la lista de reglas en el **firewall** e indica cuáles se están cumpliendo y cuántas veces se cumplen como resultado de esta comunicación, tanto en la tabla nat como en la tabla filter.

     Se están cumpliendo las reglas

     – iptables -t filter -A FORWARD -i eth0 -o eth2 -j ACCEPT
     – iptables -t filter -A FORWARD -i eth2 -o eth0 -m state –state RELATED,ESTABLISHED -j ACCEPT

     2 veces la primera y 1 vez la segunda, mientras que en la tabla nat no se cumple ninguna.

```
firewall:~# iptables -t filter -L -v -n
Chain INPUT (policy DROP 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 ACCEPT     all  --  eth0   *       0.0.0.0/0            0.0.0.0/0
    0     0 DROP       all  --  eth1   *       0.0.0.0/0            0.0.0.0/0           state NEW

Chain FORWARD (policy DROP 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
    2   112 ACCEPT     all  --  eth0   eth2    0.0.0.0/0            0.0.0.0/0
    1    60 ACCEPT     all  --  eth2   eth0    0.0.0.0/0            0.0.0.0/0           state RELATED,ESTABLISHED
    0     0 ACCEPT     udp  --  *      *       0.0.0.0/0            10.209.0.40         udp dpt:7
    0     0 LOG        udp  --  *      *       0.0.0.0/0            10.209.0.40         udp dpt:7 LOG flags 0 level 4 prefix `echo'
    0     0 ACCEPT     udp  --  *      *       0.0.0.0/0            10.209.0.50         udp dpt:13
    0     0 LOG        udp  --  *      *       0.0.0.0/0            10.209.0.50         udp dpt:13 LOG flags 0 level 4 prefix `daytime'
    0     0 ACCEPT     tcp  --  *      *       0.0.0.0/0            10.209.0.40         tcp dpt:7
    0     0 LOG        tcp  --  *      *       0.0.0.0/0            10.209.0.40         tcp dpt:7 LOG flags 0 level 4 prefix `tcp'
    0     0 DROP       all  --  eth1   eth0    0.0.0.0/0            0.0.0.0/0           state NEW

Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
```

3. Comunicación entre **pc7** y servidor de UDP en el puerto 7 de **pc4**:

- Ejecuta el script de configuración para que se reinicien los contadores.
- En el escenario se encuentra lanzado en **pc4** un servidor UDP en el puerto 7 (*echo*). Realiza una captura de tráfico en **pc4(eth0)** (iptables-15.cap). Prueba a lanzar un cliente con nc desde **pc7** para que se comunique con este servidor.
- Consulta la lista de reglas en el **firewall** e indica cuáles se están cumpliendo y cuántas veces se cumplen como resultado de esta comunicación.

  Se estan cumpliendo las reglas

  – iptables -t filter -A FORWARD -i eth2 -o eth1 -d 100.209.0.40 -p udp –dport 7 -j LOG –log-prefix echo
  – iptables -t filter -A FORWARD -i eth2 -o eth1 -d 100.209.0.40 -p udp –dport 7 -j ACCEPT

  1 vez cada una, ya que la primera solo es el Log.

```
firewall:~# iptables -t filter -L -v -n
Chain INPUT (policy DROP 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 ACCEPT     all  --  eth0   *       0.0.0.0/0            0.0.0.0/0
    0     0 DROP       all  --  eth1   *       0.0.0.0/0            0.0.0.0/0           state NEW

Chain FORWARD (policy ACCEPT 1 packets, 33 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 ACCEPT     all  --  eth0   eth2    0.0.0.0/0            0.0.0.0/0
    0     0 ACCEPT     all  --  eth2   eth0    0.0.0.0/0            0.0.0.0/0           state RELATED,ESTABLISHED
    1    33 LOG        udp  --  eth2   eth1    0.0.0.0/0            100.209.0.40        udp dpt:7 LOG flags 0 level 4 prefix `echo'
    1    33 ACCEPT     udp  --  eth2   eth1    0.0.0.0/0            100.209.0.40        udp dpt:7
    0     0 ACCEPT     udp  --  eth2   eth1    0.0.0.0/0            100.209.0.50        udp dpt:13
    0     0 LOG        udp  --  eth2   eth1    0.0.0.0/0            100.209.0.50        udp dpt:13 LOG flags 0 level 4 prefix `daytime'
    0     0 ACCEPT     tcp  --  eth0   eth1    0.0.0.0/0            100.209.0.40        tcp dpt:7
    0     0 LOG        tcp  --  eth0   eth1    0.0.0.0/0            100.209.0.40        tcp dpt:7 LOG flags 0 level 4 prefix `tcp'
    0     0 DROP       all  --  eth1   eth0    0.0.0.0/0            0.0.0.0/0           state NEW

Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
```

- Incluye en la memoria los mensajes que esta comunicación ha provocado en el fichero de log.

  Ver el log 5.1.

4. Comunicación entre **pc6** y servidor de UDP en el puerto 13 de **pc5**:

   - Ejecuta el script de configuración para que se reinicien los contadores.
   - En el escenario se encuentra lanzado en **pc5** un servidor UDP en el puerto 13 (*daytime*). Realiza una captura de tráfico en **pc5(eth0)** (iptables-16.cap). Prueba a lanzar un cliente con **nc** desde **pc6** para que se comunique con este servidor.
   - Consulta la lista de reglas en el **firewall** e indica cuáles se están cumpliendo y cuántas veces se cumplen como resultado de esta comunicación.

     Se estan cumpliendo las reglas

- iptables -t filter -A FORWARD -i eth2 -o eth1 -d 100.209.0.50 -p udp –dport 13 -j LOG –log-prefix daytime
- iptables -t filter -A FORWARD -i eth2 -o eth1 -d 100.209.0.50 -p udp –dport 13 -j ACCEPT

1 vez cada una, ya que la primera solo es el Log.

```
firewall:~# iptables -t filter -L -v -n
Chain INPUT (policy DROP 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 ACCEPT     all  --  eth0   *       0.0.0.0/0            0.0.0.0/0
    0     0 DROP       all  --  eth1   *       0.0.0.0/0            0.0.0.0/0           state NEW

Chain FORWARD (policy ACCEPT 1 packets, 54 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 ACCEPT     all  --  eth0   eth2    0.0.0.0/0            0.0.0.0/0
    0     0 ACCEPT     all  --  eth2   eth0    0.0.0.0/0            0.0.0.0/0           state RELATED,ESTABLISHED
    0     0 LOG        udp  --  eth2   eth1    0.0.0.0/0            100.209.0.40        udp dpt:7 LOG flags 0 level 4 prefix `echo'
    0     0 ACCEPT     udp  --  eth2   eth1    0.0.0.0/0            100.209.0.40        udp dpt:7
    1    32 LOG        udp  --  eth2   eth1    0.0.0.0/0            100.209.0.50        udp dpt:13 LOG flags 0 level 4 prefix `daytime'
    1    32 ACCEPT     udp  --  eth2   eth1    0.0.0.0/0            100.209.0.50        udp dpt:13
    0     0 LOG        tcp  --  eth0   eth1    0.0.0.0/0            100.209.0.40        tcp dpt:7 LOG flags 0 level 4 prefix `tcp'
    0     0 ACCEPT     tcp  --  eth0   eth1    0.0.0.0/0            100.209.0.40        tcp dpt:7
    0     0 DROP       all  --  eth1   eth0    0.0.0.0/0            0.0.0.0/0           state NEW

Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
```

- Incluye en la memoria los mensajes que esta comunicación ha provocado en el fichero de log.

  Ver el log 5.2.

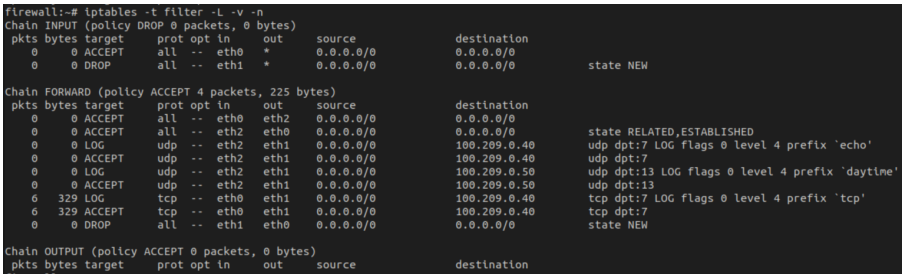
5. Comunicación entre **pc1** y servidor de TCP en el puerto 7 de **pc4**:

   - Ejecuta el script de configuración para que se reinicien los contadores.
   - En el escenario se encuentra lanzado en **pc4** un servidor TCP en el puerto 7 (*echo*). Realiza una captura de tráfico en **pc4(eth0)** (iptables-17.cap). Prueba a lanzar un cliente con **nc** desde **pc1** para que se conecte con este servidor.
   - Consulta la lista de reglas en el **firewall** e indica cuáles se están cumpliendo y cuántas veces se cumplen como resultado de esta comunicación. Es importante que observes que las reglas de la tabla **filter**, si se cumple la condición, se aplican con cada paquete que atraviesa el **firewall** y este comportamiento es diferente a lo que ocurría con las reglas de la tabla **nat**.

     Se estan cumpliendo las reglas

     - iptables -t filter -A FORWARD -i eth0 -o eth1 -d 100.209.0.40 -p tcp –dport 7 -j LOG –log-prefix tcp
     - iptables -t filter -A FORWARD -i eth0 -o eth1 -d 100.209.0.40 -p tcp –dport 7 -j ACCEPT

6 veces cada una, ya que la primera solo es el Log. Como es una comunicación Tcp por cada intercambio de mensajes las reglas se cumplen 2 veces, al contrario que en la tabla nat, donde solo se aumenta el contador si pasan más de 5 días entre cada comunicación, es decir, en el ejemplo anterior solo se habría cumplido 1 norma 1 vez en vez de las 6 de la tabla filter.



```
firewall:~# iptables -t filter -L -v -n
Chain INPUT (policy DROP 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 ACCEPT     all  --  eth0   *       0.0.0.0/0            0.0.0.0/0
    0     0 DROP       all  --  eth1   *       0.0.0.0/0            0.0.0.0/0           state NEW

Chain FORWARD (policy ACCEPT 4 packets, 225 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 ACCEPT     all  --  eth0   eth2    0.0.0.0/0            0.0.0.0/0
    0     0 ACCEPT     all  --  eth2   eth0    0.0.0.0/0            0.0.0.0/0           state RELATED,ESTABLISHED
    0     0 LOG        udp  --  eth2   eth1    0.0.0.0/0            100.209.0.40        udp dpt:7 LOG flags 0 level 4 prefix `echo'
    0     0 ACCEPT     udp  --  eth2   eth1    0.0.0.0/0            100.209.0.40        udp dpt:7
    0     0 LOG        udp  --  eth2   eth1    0.0.0.0/0            100.209.0.50        udp dpt:13 LOG flags 0 level 4 prefix `daytime'
    0     0 ACCEPT     udp  --  eth2   eth1    0.0.0.0/0            100.209.0.50        udp dpt:13
    6   329 LOG        tcp  --  eth0   eth1    0.0.0.0/0            100.209.0.40        tcp dpt:7 LOG flags 0 level 4 prefix `tcp'
    6   329 ACCEPT     tcp  --  eth0   eth1    0.0.0.0/0            100.209.0.40        tcp dpt:7
    0     0 DROP       all  --  eth1   eth0    0.0.0.0/0            0.0.0.0/0           state NEW

Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
```

- Incluye en la memoria los mensajes que esta comunicación ha provocado en el fichero de log.

  Ver el log 5.3.

4 Scripts

Listing 4.1: fw1.sh

```
#!/bin/sh

iptables -t nat -F
iptables -t nat -Z

iptables -t nat -A POSTROUTING -s 10.209.0.0/16 -o eth2 -j SNAT --to-source 100.209.1.100
```

Listing 4.2: fw2.sh

```
#!/bin/sh

iptables -t nat -F
iptables -t nat -Z

iptables -t nat -A PREROUTING -i eth2 -d 100.209.1.100 -p tcp --dport 80 -j DNAT --to-destination 10.209.2.30:80

# Parte 2.2.2
iptables -t nat -A PREROUTING -i eth2 -d 100.209.1.100 -p udp --dport 5001 -j DNAT --to-destination 10.209.0.10:5001
iptables -t nat -A PREROUTING -i eth2 -d 100.209.1.100 -p udp --dport 5002 -j DNAT --to-destination 10.209.0.20:5001
```

Listing 4.3: fw3.sh

```
#!/bin/sh

# Parte 2.1
iptables -t nat -F
iptables -t nat -Z

iptables -t nat -A POSTROUTING -s 10.209.0.0/16 -o eth2 -j SNAT --to-source 100.209.1.100

# Parte 3
iptables -t filter -F
iptables -t filter -Z

iptables -t filter -P INPUT DROP
iptables -t filter -P FORWARD DROP
iptables -t filter -P OUTPUT ACCEPT

iptables -t filter -A INPUT -i eth0 -j ACCEPT

iptables -t filter -A FORWARD -i eth0 -o eth2 -j ACCEPT
iptables -t filter -A FORWARD -i eth2 -o eth0 -m state --state RELATED,ESTABLISHED -j ACCEPT

iptables -t filter -A FORWARD -i eth2 -o eth1 -d 100.209.0.40 -p udp --dport 7 -j LOG --log-prefix echo
iptables -t filter -A FORWARD -i eth2 -o eth1 -d 100.209.0.40 -p udp --dport 7 -j ACCEPT

iptables -t filter -A FORWARD -i eth2 -o eth1 -d 100.209.0.50 -p udp --dport 13 -j LOG --log-prefix daytime
iptables -t filter -A FORWARD -i eth2 -o eth1 -d 100.209.0.50 -p udp --dport 13 -j ACCEPT

iptables -t filter -A FORWARD -i eth0 -o eth1 -d 100.209.0.40 -p tcp --dport 7 -j LOG --log-prefix tcp
iptables -t filter -A FORWARD -i eth0 -o eth1 -d 100.209.0.40 -p tcp --dport 7 -j ACCEPT

iptables -t filter -A FORWARD -i eth1 -o eth0 -m state --state NEW -j DROP
iptables -t filter -A INPUT -i eth1 -m state --state NEW -j DROP
```

5 Logs

Listing 5.1: Log 3.3

```
Feb 24 09:53:06 firewall kernel: echoIN=eth2 OUT=eth1 SRC=100.209.6.70 \
DST=100.209.0.40 LEN=33 TOS=0x00 PREC=0x00 TTL=61 ID=21176 DF PROTO=UDP SPT=6666 DPT=7 LEN=13
```

Listing 5.2: Log 3.4

```
Feb 24 09:57:43 firewall kernel: daytimeIN=eth2 OUT=eth1 SRC=100.209.5.60 \
DST=100.209.0.50 LEN=32 TOS=0x00 PREC=0x00 TTL=61 ID=62502 DF PROTO=UDP SPT=6666 DPT=13 LEN=12
```

Listing 5.3: Log 3.5

```
Feb 24 10:00:58 firewall kernel: tcpIN=eth0 OUT=eth1 SRC=10.209.0.10 DST=100.209.0.40 \
LEN=60 TOS=0x00 PREC=0x00 TTL=62 ID=25373 DF PROTO=TCP SPT=6666 DPT=7 WINDOW=5840 RES=0x00 SYN URGP=0
Feb 24 10:00:58 firewall kernel: tcpIN=eth0 OUT=eth1 SRC=10.209.0.10 DST=100.209.0.40 \
LEN=52 TOS=0x00 PREC=0x00 TTL=62 ID=25374 DF PROTO=TCP SPT=6666 DPT=7 WINDOW=2920 RES=0x00 ACK URGP=0
Feb 24 10:01:10 firewall kernel: tcpIN=eth0 OUT=eth1 SRC=10.209.0.10 DST=100.209.0.40 \
LEN=61 TOS=0x00 PREC=0x00 TTL=62 ID=25375 DF PROTO=TCP SPT=6666 DPT=7 WINDOW=2920 RES=0x00 ACK PSH URGP=0
Feb 24 10:01:10 firewall kernel: tcpIN=eth0 OUT=eth1 SRC=10.209.0.10 DST=100.209.0.40 \
LEN=52 TOS=0x00 PREC=0x00 TTL=62 ID=25376 DF PROTO=TCP SPT=6666 DPT=7 WINDOW=2920 RES=0x00 ACK URGP=0
Feb 24 10:01:18 firewall kernel: tcpIN=eth0 OUT=eth1 SRC=10.209.0.10 DST=100.209.0.40 \
LEN=52 TOS=0x00 PREC=0x00 TTL=62 ID=25377 DF PROTO=TCP SPT=6666 DPT=7 WINDOW=2920 RES=0x00 ACK FIN URGP=0
Feb 24 10:01:18 firewall kernel: tcpIN=eth0 OUT=eth1 SRC=10.209.0.10 DST=100.209.0.40 \
LEN=52 TOS=0x00 PREC=0x00 TTL=62 ID=25378 DF PROTO=TCP SPT=6666 DPT=7 WINDOW=2920 RES=0x00 ACK URGP=0
```